# Technical Report

Department of Computer Science
and Engineering
University of Minnesota
4-192 EECS Building
200 Union Street SE
Minneapolis, MN 55455-0159 USA

TR 01-022

A Comparative Study on Web Prefetching

Ajay Bhushan Pandey, Ranga R. Vatsavai, Xiaobin Ma, Jaideep Srivastava, and Shashi Shekhar

May 31, 2001

# A Comparative Study of Web Prefetching Algorithms

Ajay B. Pandey* Ranga R. Vatsavai* Xiaobin Ma*
Jaideep Srivastava* Shashi Shekhar*

May 9, 2001

**Abstract**

The growth of the World Wide Web has emphasized the need for improved user latency. Increasing use of dynamic pages, frequent changes in site structure, and user access patterns on the internet have limited the efficacy of caching techniques and emphasized the need for prefetching. Since prefetching increases bandwidth, it is important that the prediction model is highly accurate and computationally feasible. It has been observed that in a web environment, certain sets of pages exhibit stronger correlations than others, a fact which can be used to predict future requests. Previous studies on predictive models are mainly based on pair interactions of pages and TOP-N approaches. In this paper we study a model based on page interactions of higher order where we exploit set relationships among the pages of a web site. We also compare the performance of this approach with the models based on pairwise interaction and the TOP-N approach. We have conducted a comparative study of these models on a real

*Department of Computer Science, University of Minnesota, Minneapolis, MN 55455, USA. Support in part by the Army High Performance Computing Research Center under the auspices of Department of the Army, Army Research Laboratory Cooperative agreement number DAAH04-95-2-0003/contract number DAAH04-95-C-0008, and by the National Science Foundation under grant 9631539. Email: {pandey, vatsavai, xiaobin, srivasta, shekhar}@cs.umn.edu

server log and five synthetic logs with varying page frequency distributions to simulate different real life web sites and identified dominance zones for each of these models. We find that the model based on higher order page interaction is more robust and gives competitive performance in a variety of situations.

Keywords: Association Rules, Top-N, Pairwise interaction, Order of statistics, Mixed order, Higher order interactions, Prefetching, Log Analysis.

# 1 Introduction

The exponential growth of the World Wide Web has caused a dramatic increase in Internet traffic and serious performance degradation in terms of user latency on the Internet. This has led to a great deal of effort devoted to studying web caching [8], [7], [10] and prefetching techniques through the use of proxies. Most web caching schemes maintain global caches, which straddle across users. Because pages use static html and hence are the same for all users, global caching works quite well. However there is an increasing trend of generating dynamic pages in response to http requests on account of personalizing customizing experience, frequent updates in databases, and secure transactions. As the Internet grows and becomes a primary means of communication, the majority of pages will tend to be dynamic pages. In such a situation, traditional web caching will be unworkable. As a result, attempts are being made to design an intelligent prefetcher based on an improved prediction model which will improve user latency with minimal increase in bandwidth.

Primarily, two techniques, i.e. caching and prefetching, are currently used for improv-

ing user latency. Several studies have been conducted on cache replacement policies for improving cache hit ratio. On the prefetching side, studies have been conducted on prefetching models based on Top-N, pairwise interaction models, association rules, and path profile analysis. Increasing use of dynamic pages, frequent changes in site structure and user access patterns on the Internet have limited the efficacy of caching techniques and emphasized the need for prefetching with a predictive model which is less expensive to build and update and which has improved predictive performance

## 1.1 Application Domain

**Intelligent Prefetching Based Proxy Server:** The application domain of a predictive prefetcher is in a proxy server, where it is used to predict a user's future accesses based on his or her past history and the server access pattern. Based on this prediction, web pages are prefetched and cached in the proxy server's cache to reduce user latencies. A block diagram of the the proxy server is shown in Figure 1.

The various modules and the sequence of operations are described below:

Server logs are filtered and preprocessed, and appropriate rules are generated based on the prediction model. This can be done online or off-line at specified intervals to generate up-to -date association rules.

There are request history list, a prefetching queue, and a prefetching thread. The prefetching queue maintains a list of the pages which need to be prefetched It is created using the history list and the prediction module. The prefetcher thread takes out an item from the prefetching queue and checks whether the page is in the cache. If the page is not

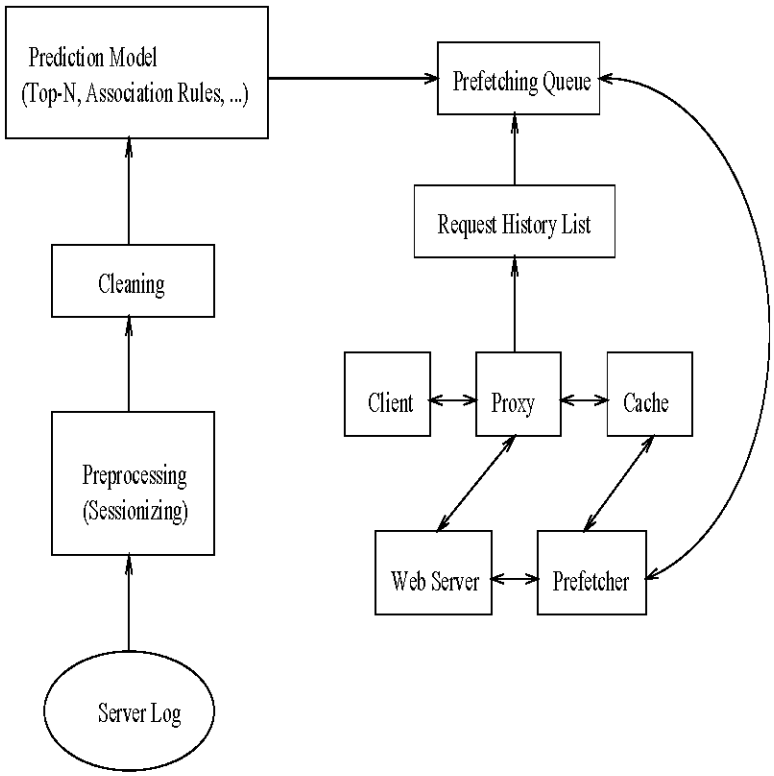



Figure 1: Proxy Server Architecture

found in the cache, it gets the page from the main server and puts it in the cache. The prefetcher thread, the prefetching queue, and the user history list are destroyed at the end of the session.

A cache is created for each user session and then deleted at the end of the user session. The proxy listens to the incoming client requests. After receiving a request, it checks whether the corresponding page is in the cache, and, if the page is found, it reads the page from the cache and sends it to the client. If the page is not found in the cache, the proxy server makes a request to the main server, gets the page from the server, and sends it to the client. It also adds the page in the cache if the page is cacheable. It adds the URL of the request in the user history list. After every request the prefetching queue is updated on the basis of the history list.

**Related work and our contribution:** Previous studies have focused on prefetching algorithms. These studies used predictive models using different orders of statistics on separate server logs with varying results. We define order of statistics by the number of pages considered together for observing interaction patterns. For example, first order statistics does not consider any page interactions, second order statistics looks at pairwise interactions (e.g. $a \rightarrow b$) and higher order statistics looks at two or more page page interactions (e.g. $\{a, b, ..\} \rightarrow \{x, y, ..\}$, i.e. $\{a_i\} \rightarrow \{x_j\}$, where $i \geq 2$ and $j \geq 2$). Often the problem is in determining which algorithm is suitable for a specific situation as there are no studies evaluating these algorithms on common logs. In this study we have made theoretical and experimental comparison of three different models i.e the Top-N, Pairwise interaction, and mixed order statistical models on a common set of logs with varying page frequencey distributions and orders of statistics. In theoretical comparison, we have derived expressions for metrices which can judge predictive efficacy of a model. Since in real server logs, we do not have complete idea of how pages are interacting with each other, it is difficult to observe and compare the precise effect of page frequencey distribution and and the order of statistics on prediction. Therefore, we generated five syntheitic logs with varying page frequency distributions and orders of statistics and observed the performance of the three models. This analysis gave us interesting insights: (a) simpler methods such as Top-N perform well in some situations, and (b) mixed order models are more robust and adaptive to variety of situations.

In Section 2 we formally define the prediction problem and optimization criteria. In Section 3 we discuss three candidate models and a brief review of the literature. In Section 4 we provide results of comparative analysis of these candidate models based on theoretical

and experimental considerations, and finally in Section 5 we present conclusions and future work.

# 2 Problem Formulation

The main objective of a proxy server is to reduce user perceived latency by prefeteching documents based on some prediction model. The success of a proxy server thus depends on how accurately the prediction model predicts a set of documents that will most likely be accessed by the user. We can formally define the problem of prefetching using prediction as follows:

**Given**

- Sessionized Web log (previous history)
- Current session

**Find**

- Predict most probable future page requests

**Objective**

- Maximize cache hit ratio
- Minimize bandwidth ratio

## 2.1 Optimization Criteria

The main objective of any prediction model is to come up with a set of documents that will most likely be accessed in the near future. Ideally, if the prediction is made in time before the user actually requests the page, and all the user requested pages are correctly predicted, prefetched and cached, then we will have zero latency and no extra bandwidth. But in reality, part of the prefetched pages may never be used, or the user requested pages may not be in the cache. Both of these situations result in increased network traffic and delays. So the performance of a prediction model can be measured in terms of cache hit ratio and bandwidth ratio. We consider these two key performance matrices for comparing the candidate prefetching algorithms.

*Bandwidth Ratio*: Bandwidth ratio is defined as the ratio of the total number of files fetched from the server and total number of total number of requests.

*Cache hit Ratio*: Cache hit ratio is defined as the ratio of the number of pages found in the cache and the total number of pages requested.

The optimization criteria is to reduce the bandwidth and increase the number of cache hits. There is a trade off between these two performance matrices. Cache hit ratio can be improved by predicting more pages, whereas bandwidth can be reduced by minimizing the number of page transfers. A good prediction model will give a high cache hit ratio for a low bandwidth ratio.

Cohen et al. [3] has studied the optimization problem in a pairwise interaction model with respect to performance metrics 'Recall', 'Precision', and 'Average Hintsize' and has proved that the problem is NP-Complete.

# 3 Related Work and Candidate Algorithms

The efficacy of prefetching and caching strategies depends upon how accurate the prediction model is. The prediction models can be categorized into two groups: one based on local knowledge of the access patterns and preferences of the client, and the other based on the server's knowledge of access patterns and popularity of its documents. In this paper we concentrated on the models based on server knowledge and its use for determining prefetching strategies. We classified prefetching models based on the order of statistics used by them. By the order of statistics we mean the number of pages being used in generating interaction statistics. The first order statistics would have information only on individual frquency of pages. Top-N model is based on this statistics. Pairwise interaction models proposed in [3], [11], [2], [14] are based on second order statistics, where page interactions between two pages are exploited. The association rules model [13] is based on mixed order of statistics because it exploits page interactions among two or more pages.

## 3.1 First Order Statistics Models

**Top-N**: In the Top-N approach [9], the server periodically calculates a list of most popular documents and makes them available to the clients, which they can prefetch. This approach

uses trace driven simulation based on access logs from various servers to evaluate Top-N prefetching. Clients can then prefetch the documents from the Top-N set.

## 3.2 Second Order Statistics i.e Pairwise Interaction Models:

Sarukkai and Ramesh [14] proposed a probabilistic model for web link prediction and path analysis using Markov chains. The Markov Chain Model consists of a matrix of state transition probabilities and initial state probability vector $\lambda$. $\hat{S}(t) = \hat{S}(t-1)A$, where $\hat{S}(t)$ and $\hat{S}(t-1)$ are probability vectors for all states at time $t$ and $t-1$ respectively and $A$ is a matrix representing transition probabilities from one state to another. If there are $n$ states, then the matrix $A$ will be of the order $n \times n$. Matrix $A$ can be estimated as follows:

$$A(s, s^{'}) = \frac{C(s, s^{'})}{\Sigma_{s^{"}} C(s, s^{"})}$$

$$\lambda(s) = \frac{C(s)}{\Sigma_{s^{'}} C(s^{'})}$$

where $C(s, s^{'})$ is the count of the number of times $s^{'}$ follows $s$ in the training data. All client requests are buffered into a client buffer and flushed once a minimum threshold is exceeded. Given a start URL, the tour generator outputs a sequence of states which corresponds to the tour generated by the model. The tour predicts a sequence of URLs which will be visited next by the client. The training and simulation were conducted on data collected at Research Triangle Park,NC. It was observed that 50% of the requests could be predicted with the Markov chain prediction model.

Padmanabhan and Mogul [11] suggested a server-knowledge-based predictive model

based on a dependency graph where there is a node for every file and an arc from node $A$ to node $B$ if $B$ was accessed within $W$ accesses after $A$ , where $W$ is the look ahead window size. Every arc is assigned a weight, which is the ratio of the number of accesses to $B$ within a window after $A$ to the number of accesses to $A$ itself. When $A$ is accessed, the server sends a hint to the client to prefetch $B$ if the arc from $A$ to $B$ has a large weight.

Azer [2] proposed a speculative model based on a probability matrix $P$, where $P[i..j]$ is the conditional probability that a document $d_j$ will be requested within a limited window of time $T_w$, given that $d_i$ has already been requested. The server log was also used to generate closure of $P$, i.e., $P^* = P^N$. When a request for a document $d_i$ is received, the server responds by sending to the client the documents $d_j$ that satisfy an inequality based on functions $P$ and $P^*$. This inequality determines the particular policy employed. For example, it could send hints for prefetching document $d_j$ along with a requested document $d_i$ if $P^*[i,j] \geq t_p$ for some threshold property $0 < t_p \leq 1$.

Cohen et al. [3] have suggested efficient algorithms for prefetching web documents based on the pairwise interaction model. For each resource $a$, the proxy constructs a volume $v_a$, which is a list of resources that are likely to be requested in the near future after an access to $a$. Also the predictions were considered between some minimum and maximum time intervals. In greedy approach, set $v_a$ is iteratively augmented by adding new resource $b$ to volume $v_a$ based on the marginal utility $u(b,a)$. An improvement over greedy heuristic is achieved through pairwise heuristic, in which volume construction is based on the probability that the resource $a$ and $b$ are accessed together, independent of the predictions generated by other resources. The probability $p_{b|a}$ is defined as the proportion of requests for resource $a$

that are followed by a request for resource $b$ by the same client within at least $T$ seconds and after $\tau$ seconds. Initial sets of volumes are generated after applying a threshold probability $p_t$ to the implication probabilities $p_{b|a}$. This volume set is then thinned by removing ineffective predictions.

## 3.3 Mixed Order Statistics Models

**Association Rules:** An association rules-based prefetching scheme was proposed in [13]. This approach is primarily based on knowledge discovery of server access patterns through data mining. The assumption is that within a site, there are certain sets of URLs which exhibit strong correlations/associations with each other. With this approach we can extend the pairwise dependencies to set dependencies. We can discover set dependencies among the pages of a site using data mining in the form of association rules and predict future accesses. Efficient algorithms for computing association rules can be found in [1], [16], [6].

The association rules correlate the presence of a set of items with another range of values for another set of variables. An association rule is an implication of the form $X \Rightarrow Y$, where $X \subset I$, $Y \subset I$, and $X \cap Y = \emptyset$. Here $I = i_1, i_2, \ldots, i_m$ is a set of items (URLs). The meaning of association rules here is that if a client visits X, then it is also likely that he will visit Y. The support for the rule $X \Rightarrow Y$ is the percentage of client sessions that hold all the pages in the union, i.e., the set $X \cap Y$. A low support will imply that there is not enough evidence that the pages in $X \cap Y$ occur together in a majority of the sessions. For computing the confidence for association rule $X \Rightarrow Y$, we take all the user sessions which contain $X$,

and the confidence of the rule is the percentage of such user sessions that also contains $Y$. Since usually such transaction databases contain extremely large amounts of data, current association rule discovery techniques try to prune the search space according to the support for the items under consideration [4].

**Path Profiles:** Schechter et al. [15] suggested using path profiles to predict HTTP requests. This study specifically targets prefetching dynamic pages. Its model is based on matching the path of the current user with that in the server database derived from the past access history. The study suggests generating a tree from the path profile consisting of paths whose frequency exceeds a certain threshold. For predicting a future path, a user's current session in the form of a path is passed and is matched with the tree.

The above literature survey is not exhaustive. Since our objective is to gain an insight into dominance zones for models using various orders of statistics, we have considered only some representative models which adequately cover different orders of statistics.

# 4 Comparison of Prefetching Algorithms

We have chosen three candidate algorithms based on the order of statistics for our comparative analysis. They are Top-N (first order), pairwise interaction (second order) and association rules (mixed order). We will not use the sequential association rules model because the association rules model adequately cover mixed order statistics.

## 4.1 Theoretical Comparison

We compare the three candidate algorithms based on the assumptions and paraters they use. First we define the following terms:

1. $w$: Window size, measured in units of time or number of page accesses, that the prefetcher tries to predict future events (i.e. pages).

2. frequency($P_i$): Total number of times the page $P_i$ is accessed (first order interaction) in the entire session log. Top-N exploits these frequencies.

3. frequency($P_i|P_j$): Total number of times the page $P_j$ is accessed in a window ($w$) after the $P_i$ is accessed in a user session (second order interaction). Pairwise interaction models exploits these frequencies.

4. frequency($P_i|P_j, j = 1, ..m$): total number of times a set of pages occur together in a window ($w$) after the page $P_i$ is accessed in a user session (higher order interaction). Association rules exploit second and higher order page interactions.

The Top-N algorithm uses the frequency($P_i$) measure to predict best N pages that are likely to be accessed in the near future given that the page $P_i$ is currently accessed. The algorithm is very simple and computationally efficient. Pairwise interaction models use the frequency($P_i|P_j$) measure to predict best N pages that are likely to be accessed in the near future given that the page $P_i$ is currently accessed [11], [14], [2], [3]. Mixed order models such as association rules and $k^{th}$ order Markov models, are more flexible and they use the frequency($P_i|P_j, j = 1, ..N$) measure for future page access prediction [12], [5]. By higher order interaction we mean the interaction between three or more variables. In some cases

pairwise interaction models may perform worse than the top-N model in terms of cache hit ratio because some logs may have strong trends in frequency($P_i$) but weak trends in frequency($P_i|P_j$). On the other hand, mixed order models are more flexible than top-N and pairwise models and the performance is also better in terms of cache hit ratio, because mixed models take advantage of all the possible interactions including first and second order. Now we define cache hit ratio and band width ratio in terms of the frequency measures defined above.

Given a sequence of sessions $S_1, S_2, \ldots, S_k$ and a page access sequence in a given session $S_i$, where $S_i = P_i^k, k = 1, \ldots, m$, we can define cache hit ratio in general terms as follows:

$$\text{Cache hit ratio(CHR)} = \frac{count_{i,j}(P_i^j | P_i^j \in \text{cache})}{count_{i,j}(P_i^j)}$$

Using the probability measures defined above, we give expressions for cache hit ratio for each of the candidate algorithms as follows:

$$CHR_{Top-N} = \frac{count_{i,j}((P_i^j | P_i^j \in f^1(N)) \vee (P_i^j \in \{P_i^k, k = 1, \ldots, j-1\}))}{count_{i,j}(P_i^j)}$$

$$CHR_{pairwise} = \frac{count_{i,j}((P_i^j | P_i^j \in \cup_{k=1}^{j-1} f^2(P_i^k, w, N)) \vee (P_i^j \in \{P_i^k, k = 1, \ldots, j-1\}))}{count_{i,j}(P_i^j)}$$

$$CHR_{mixed} = \frac{count_{i,j}((P_i^j | P_i^j \in \cup_{k=1}^{j-1} f^m(\{P_i^l, l = 1, \ldots, k\}, w, N)) \vee (P_i^j \in \{P_i^k, k = 1, \ldots, j-1\}))}{count_{i,j}(P_i^j)}$$

The function $f^m$ returns a set of predicted pages depending upon the chosen prediction model. The function $f^m$ is defined in terms of the corresponding page access counts (frequencies) defined above and 'm' refers to the order of the statisticsl. The expression for band width ratio is as following:

$$BWR_{Top-N} = \frac{count_{i,j}(P_i^j | P_i^j \ni f^1(N) \cup \{P_i^k, k = 1, \ldots, j-1\})) + \Sigma_{i,j} size(f^1(N))}{count_{i,j}(P_i^j)}$$

$$BWR_{pairwise} = \frac{count_{i,j}(P_i^j | P_i^j \ni f^2(P_i^k, w, N) \cup \{P_i^k, k = 1, \ldots, j-1\} + \Sigma_{i,j} size(\cup_{k=1}^{j-1} f^2(P_i^k, w, N))}{count_{i,j}(P_i^j)}$$

$$BWR_{mixed} = \frac{A + B}{count_{i,j}(P_i^j)}$$

where

$$A = count_{i,j}(P_i^j | P_i^j \ni f^m(\{P_i^l, l = 1, \ldots, k\}, w, N)) + \cup\{P_i^k, k = 1, \ldots, j-1\}$$

and

$$B = \Sigma_{i,j} size(\cup_{k=1}^{j-1} f^m(\{P_i^l, l = 1, \ldots, k\}, w, N))$$

## 4.2 Experimental Comparison

In earlier studies,the above models were applied on different server logs with varying results. But the question has always been that how these models will perform and compare against one other and whether we can identify certain characteristics of a server log which favor one model against another and obtain the dominance zone for these models. In this experiment, we created five synthetic logs with different characteristics and orders of interaction to see how these factors affect the predictive efficacy of a model. In addition, we also used a real server log from an e-commerce site. We compared the performance of these algorithms based on cache hit ratio and band width ratio measures.

## 4.3 Experiment Design

A Block diagram for the experimental setup is shown in figure 2. We generated five different synthetic logs each with 20,000 sessions containing over 200,000 requests to test the models.The total number of pages was 4000 in each of these synthetic logs. We varied the number of rules, symmetry of rules and page frequency distribution in these logs and sessionized them.Using the similar parameters, we generating five training logs for training the models. We used a proxy server described in figure 1 to fetch client request from the test logs and to prefetch on the basis of the predictive model.

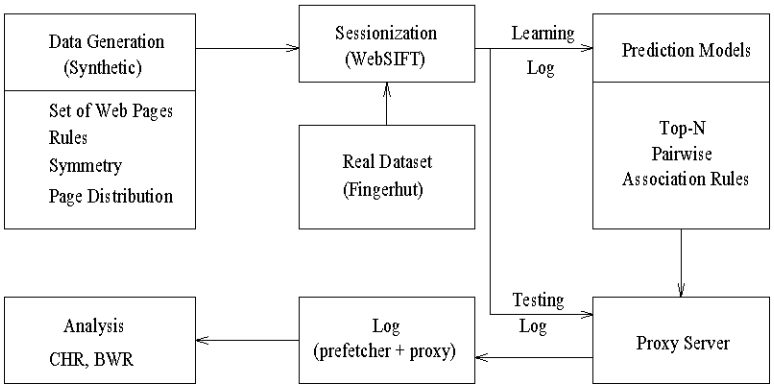



Figure 2: Block diagram for experimental setup

## 4.4 Logs

The characteristics of each of the logs are described below:

**Data Set 1(DS1) - Uniform Distribution:** This log contains 20 association rules of size 4 (A,B,C,D) having support 1% and confidence 50%. Each session contains on an average 15 requests. The remaining requests in the sessions are uniformly distributed over all the pages (0,4000). The four pages in the rules always appear in the same sequence in a session. Figure 3(a) shows the page distribution for this log. This data set simulates a web site where 2% of the pages have a strong association with each other and the remaining pages are randomly accessed. This log has predominance of second and higher order interactions because it has association rules of length 4 with significant support.

**Data set 2(DS2):** This dataset is similar to Data Set 1 except that the items have been shuffled so the frequent pages do not occur in the same sequence. This log too has a predominance of higher order interactions. Figure 3(b) shows the page distribution for this log.

**Data Set 3(DS3):** This log has 100 association rules of size 2 (A, B) with 1% support and

100% confidence. Each session contains an average of 6 requests. The remaining requests in the sessions are uniformly distributed aver the rest of the pages. Since this log has rules of size 2 only, it has a predominance of second order interactions The two pages of the rules always occur in the same temporal order. Figure 3(c) shows the page distribution for this log.

**Dataset 4(DS4):** This log has 20 association rules of size 4 ( A,B,C,D) with 1% support and 50% confidence. Each session contains an average of 15 requests. The remaining requests in the sessions are from a group of select pages. The page distribution is shown in Figure 3(d). This log simulates a web site with 20 mutually exclusive large items sets of four pages each and the rest of the request are from a select group of pages.The pages falling in this group generate some more association rules. This log is a mix of first, second and higher order interactions because it has a predominance of some select pages as well as association rules with significant support.

**DataSet 5 (DS5) Skewed Distribution:** Each session contains an average of 15 requests. Figure 3(e) describes the frequency distribution of the pages. This data set simulates a web site where page distributions are highly skewed.This log has a predominance of first order interactions because of skewed page distribution. Skewed distribution also generates second and higher order interactions.

**Data set 6(DS6):** This is a real server log from an E-commerce site. It has 8,147 pages and 19,430 sessions having 243,039 requests. The log was cleaned using WebSIFT [17] system and then broken down into a series of sessions. The page distribution over the sessions is shown in figure 3(f). The figure shows that the page distribution is highly skewed. Also

this log has generated association rules with good support and confidence; we can say that this log also has an element of higher order interactions also.

The methods of training and prefetching for each of these models are briefly described below:

**TOP-N model:** On training, this model generated a list of pages sorted by their usage count for each of the training logs. During testing, proxy prefetched the Top-N pages at the begining of each session and kept them in the cache. We measured the Cache Hit Ratio and Bandwidh ratio for differnt vaues of N.

**Pairwise interaction model:** Each training log was scanned to created pair counters (i.e., $c(a, b)$ which was incremented when page $b$ was acccessed within next $w$ accesses after page $a$ was accessed). In this experiemnt we fixed $w$ to be the size of session. During testing, after each request to a page $a$, an $n$ number of pages $\{x_i, i = 1, \ldots, n\}$ sorted by the page counter values $c(a, x_i)$ are prefetched. Cache hit ratio and band width ratio were measured by varying $n$.

**Assocation rules model:** We used apriori algorithm to mine assocation rules from each of the the training logs for the various levels of support and confidence. The procedure for prefetching is described in [8]. We also varied number of pages which would be prefectched from the rules. We measured cache Hit Ratio and Bandwidth ratio by varying support, confidence and number of pages prefetched.

## 4.5 Observations

**Results (DS1):** Figure 4 shows the comparative performance of the three models. As can be seen, the association rules model gives a significantly higher Cache Hit Ratio (CHR) vs Bandwidth Ratio(BWR) performance as compared to the pairwise interaction and Top-N models. This server log has a predominance of higher order interactions, which seems to favor the association rules model.

**Results (DS2):** For this log, too, the results (Figure 5) are similar to those in DS1. We observe that the association rules model's performance is significantly higher as compared to the pairwise interaction and TopN models, which is probably due to the prevalence of higher order interactions.

**Results (DS3):** Figure 6 shows the comparative performance of the three models. As can be seen, the association rules and the pairwise model give much higher Cache Hit Ratio for a range of bandwidth Ratio as compared to the TOP-N model. We also observe that unlike in DS1 and DS2, the pairwise interaction model performs very close to the association rules model. This log has a prevalence of second order interactions which probably favors pairwise interaction and the association rules model. Also it is expected that with smaller session sizes and smaller differences between the rule size and the session size, the pairwise model performance would converge to that of the association rules model provided the pages in the rules appear in the same order in the sessions.

**Results (DS4):** We find here (Figure 7) that, all the three models gives comparable performance. Since the distribution is highly skewed, and has a prevalence of the first, second and higher order interactions, all three models are competitive.

**Results (DS5):** Figure 8 compares the performance of the three models for this log. We see that the performance of all three models is comparable. Here also since the page distribution is highly skewed in this log, it has a predominance of the first, second and higher order interactions, which makes all the three models competitive.

**Results (DS6):** In this (Figure 9) case we find that the pairwise interaction model's performance is higher than those of the other two models. This server log exhibits very strict pairwise sequential associations (i.e., second order interactions) which seems to result in higher performance for the pairwise interaction model. Since the page distribution is skewed, it has significant first and higher order interactions too (but not as many second order interactions), which results in good performance for the Top-N and association rules models too.

## 4.6 Summary of Results

We observe that the association rule model which is based on mixed order gives competitive performance in the terms of cache hit ratio and band width ratio because it captures all orders of statistics. Even where the page frequency distribution is highly skewed and does not ostensibly contains any association rules, the skewed distribution itself throws some rules which makes the mixed order statistics model win. The converse is obvously not true. TOP-N model would not be suitable for a log which has large number of higher order interaction rules involving large number of page. A mixed order model will be more suitable for such logs. Similarly all second orders of interactions exploited by the pair interaction models are also captured by the mixed order model making it more versatile and robust.

# 5 Conclusion and Future work

In this paper we have studied and used three candidate algorithms for predicting future page accesses based on past history. We compared them on a common framework in terms of frequency measures and the order of the model. We tested these algorithms on common simulated and real web logs in order to identify the dominance zones. From our experiments, we found that Top-N performed well in cases where page frequency distribution is highly skewed but involves only few pages. Mixed models are more robust and performed well in a variety of situations. They also offer more free parameters to optimize than Top-N and pairwise interaction models.

We are currently working on the expressions for the cache hit ratio and the band width ratio to provide formal lemmas comparing various models and expecting results in the near future.

# References


[1] Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules. In *Proc. of the 20th Int'l Conference on Very Large Databases*, September 1994.

[2] Azer Bestavros. Using speculation to reduce server load and service time on the www. In *In proceedings of CIKM'95: The Fourth ACM International*, 1995.

[3] Edith Cohen, Balachander Krishnamurthy, and Jennifer Rexford. Efficient algorithms for predicting requests to web servers. In *Proc. IEEE INFOCOM*, March 1999.

[4] R. Cooley, B. Mobasher, and J. Strivastava. Web mining: Information and pattern discovery on the world wide web. In *In Proccedings of the 9th IEEE International Conference on Tools with Artificiat Intelligence (ICTAI'97)*, November 1997.

[5] Mukund Deshpande and George Karypis. Selective markov models for predicting web-page accesses. Technical Report No. TR-00-

056, Department of Computer Science, University of Minnesota, 2000. http:/www.cs.umn.edu/ karypis/publications/Papers/PDF/select.pdf.

[6] J. Han and J. Pei. Mining frequent patterns by pattern-growth: Methodology and implications. In *ACM SIGKDD Explorations (Special Issue on Scaleble Data Mining Algorithms)*, December 2000.

[7] B. Krishnamurthy and C.E. Wills. Piggyback server invalidation for proxy cache coherency. In *Proc. World Wide Web Conference*, April 1998.

[8] T. M. Kroeger, D.E .Long, and J.C.Mogul. Exploring the bounds of web latency reduction from caching and prefetching. In *Proc. USENIX Symp. on Internet Technologies and Systems*, December 1997.

[9] Evangelos P. Markatos and Catherine E. Chironaki. A top 10 approach for prefetching the web. In *In proceedings of INET'98: Internet Global Summit*, July 1998.

[10] J.C. Mogul. Hinted caching in the web. In *Proc. ACM SIGOPS European Workshop*, 1996.

[11] Venkata N. Padmanabhan and Jeffrey C. Mogul. Using predictive prefetching to improve world wide web latency. *Computer Communication Review*, 26, 1996.

[12] Ajay Pandey, Jaideep Srivastava, and Shashi Shekhar. A web intelligent prefetcher for dynamic pages using association rules - a summary of results. Workshop on Web Mining, First SIAM International Conference on Data Mining (SDM 2001).

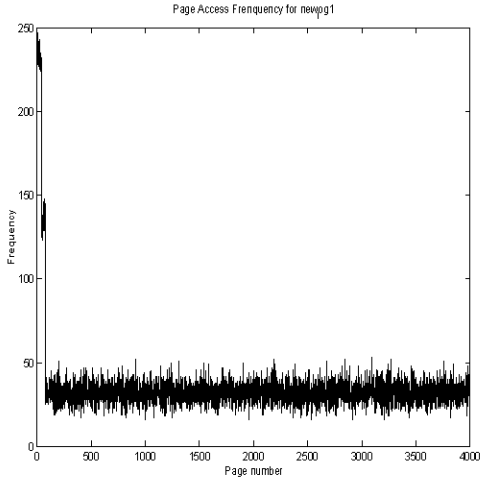
[13] Ajay Pandey, Jaideep Srivastava, and Shashi Shekhar. Web proxy server with an intelligent prefetcher for dynamic pages using association rules. Technical Report No. TR-01-004, Department of Computer Science, University of Minnesota, January 2001. http:/www.cs.umn.edu/ pandey/assoc.pdf.

[14] Sarukkai and R. Ramesh. Link prediction and path analysis using markov chains. In *In proceedings of 9th World Wide Wide Conference*, 2000.

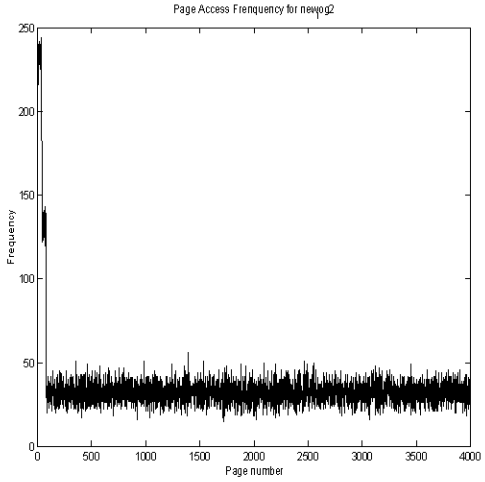
[15] Stuart Schechter, Murali Krishnan, and Michael D. Smith. Using path profiles to predict http requests. In *WWW 7*. http://decweb.ethz.ch/WWW7/1917/com1917.htm, 1998.

[16] Ramakrishnan Srikant and Rakesh Agrawal. Mining generalized association rules. In *Proc. of the 21st Int'l Conference on Very Large Databases*, September 1995.

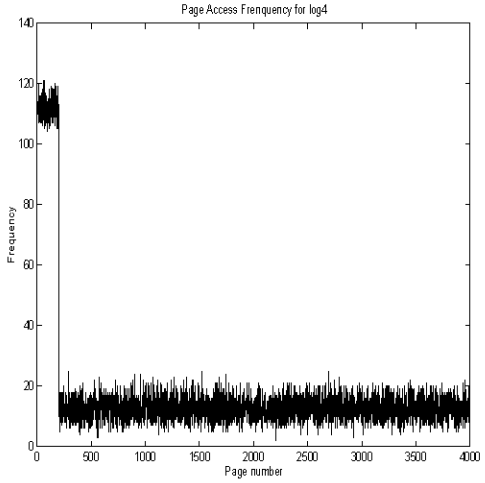
[17] WebSIFT. Web site information filter project. University of Minnesota, 2000, http://www.cs.umn.edu/Research/websift/.



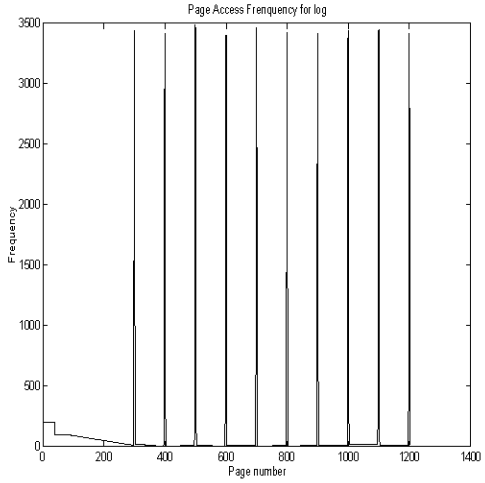

(a) Page Distribution for DS1.



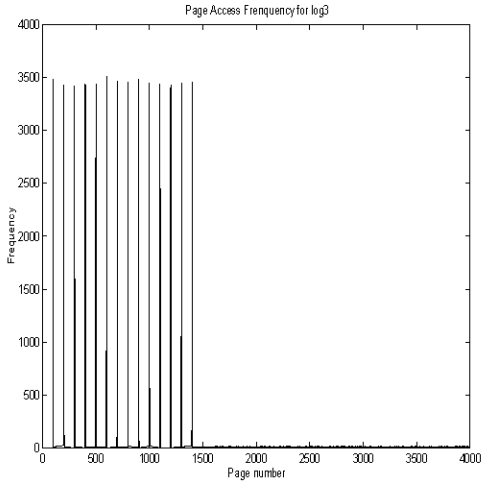

(b) Page Distribution for DS2.



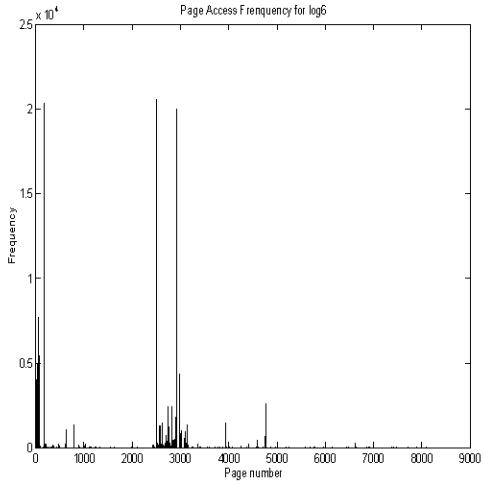

(c) Page Distribution for DS3.




(d) Page Distribution for DS4.




(e) Page Distribution for DS5.




(f) Page Distribution for DS6.



Figure 3: Page distribution graphs for various logs

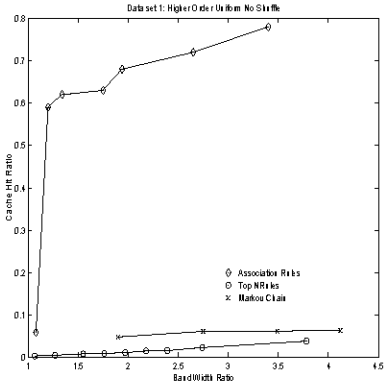



Figure 4: CHR vs. BWR for DS1.

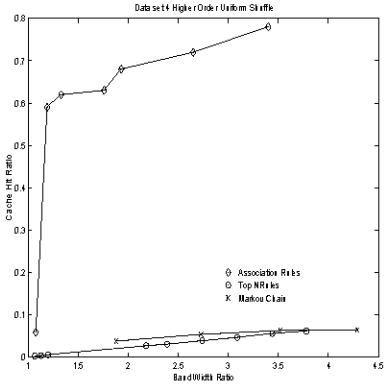



Figure 5: CHR vs. BWR for DS2.

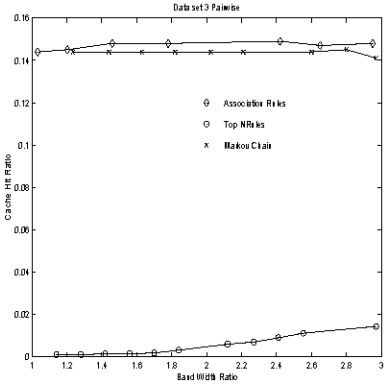



Figure 6: CHR vs. BWR for DS3.

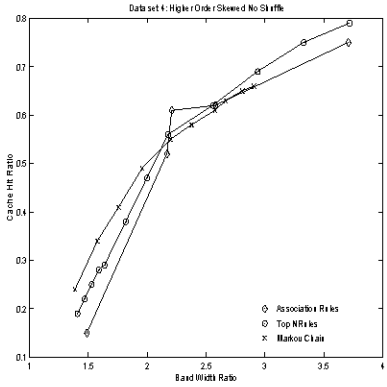



Figure 7: CHR vs. BWR for DS4.

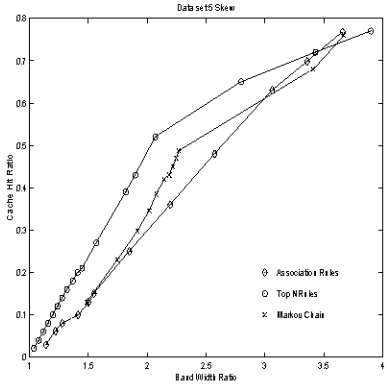



Figure 8: CHR vs. BWR for DS5.

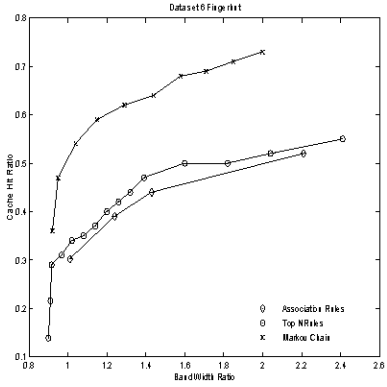



Figure 9: CHR vs. BWR for DS6.